AMENDMENTS TO THE CLAIMS

Claims 1-25 are pending. Claims 1, 8, 11, 17, 21 and 22 are amended. Dependent claims 23-25 are newly added. The remaining claims are unchanged.

The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for acquiring access to an object in an object-based system, the method comprising:

providing a thread that attempts to acquire access to an object;

identifying a memory address value associated with the object;

identifying a first synchronization construct that is suitable for use in granting access to the object;

determining **[[when]]** **that** the first synchronization construct is **un**available;

when **it is determined that** the first synchronization construct is unavailable, determining **[[whether]]** **that** the thread that is attempting to acquire access to the object is already associated with the object;

when **it is determined that** the thread is already associated with the object, releasing the association **such that the thread is not associated with the object**~~**; and**~~

~~**associating the first synchronization construct with the object when it is determined that the first synchronization construct is available**~~.

2. (Previously Presented) A method for acquiring access to an object as recited in claim 1 wherein the first synchronization construct is identified using at least part of the memory address value, and wherein identifying a first synchronization construct that is suitable for granting access to the object includes:

indexing into a data structure using the at least part of the memory address value, the data structure being arranged to associate a plurality of synchronization constructs with information relating to address locations in the object-based system, the plurality of synchronization constructs including the first synchronization construct, wherein at least one of the plurality of synchronization constructs is associated with more than one address location.

3. (Original) A method for acquiring access to an object as recited in claim 2 wherein identifying a first synchronization construct that is suitable for use granting access to the object further includes:

obtaining a hash value from the memory address value, wherein indexing into the data structure using the at least part of the memory address value includes indexing into the data structure using the hash value.

4. (Original) A method for acquiring access to an object as recited in claim 3 wherein the information relating to address locations includes hash values of the address locations and the data structure is a hash table.

5. (Original) A method for acquiring access to an object as recited in claim 4 wherein the first synchronization construct is a lock and at least one of the plurality of synchronization constructs is the lock, and associating the first synchronization construct with the object includes locking the object using the first synchronization construct.

6. (Original) A method for acquiring access to an object as recited in claim 1 wherein the first synchronization construct is a lock, and associating the first synchronization construct with the object includes locking the object using the first synchronization construct.

7. (Original) A method for acquiring access to an object as recited in claim 6 wherein the lock is a non-nestable, global lock.

8. (Currently Amended) An object-based computing system, the object-based computing system comprising:

a memory, the memory including a plurality of address locations;

at least one thread;

a plurality of objects, the plurality of objects including at least one object which is accessible to the at least one thread, wherein each of the plurality of objects is associated with a corresponding address of the plurality of address locations;

at least one lock, the at least one lock being accessible to the at least one object and being held by at least one locking thread, wherein the at least one lock can be released **such that the object is not locked by the at least one locking thread** in response to a determination that the at least one locking thread is the same thread as the at least one thread; and

a data structure, the data structure being arranged to associate the at least one lock with at least one corresponding address location selected from the plurality of address locations, wherein the data structure is arranged to be accessed by the at least one thread using a first value associated with the at least one object to identify the at least one lock.

9. (Original) An object-based computing system according to claim 8 further including:

a hashing mechanism, the hashing mechanism being arranged to create a hash value using information associated with the at least one object, wherein the first value is the hash value.

10. (Original) An object-based computing system according to claim 9 wherein the at least one corresponding address location is a hash value of the at least one corresponding address location.

11. (Currently Amended) A computer product for acquiring access to an object in an object-based system, the computer product comprising:

computer code for providing a thread that attempts to acquire access to an object;

computer code for identifying a memory address value associated with the object;

computer code for identifying a first synchronization construct that is suitable for use in granting access to the object, wherein the first synchronization construct is identified using at least part of the memory address value;

computer code for determining when the first synchronization construct is available;

computer code for, when the first synchronization construct is unavailable, determining whether the thread that is attempting to acquire access to the object is already associated with the object;

computer code for, when the thread **that is attempting to acquire access to the object** is already associated with the object, releasing the association **such that the thread is not associated with the object**;

computer code for associating the first synchronization construct with the object when it is determined that the first synchronization construct is available; and

a computer-readable medium that stores the computer codes.

12. (Original) A computer program product for acquiring access to an object as recited in claim 11 wherein the computer code for identifying a first synchronization construct that is suitable for granting access to the object includes:

computer code for indexing into a data structure using the at least part of the memory address value, the data structure being arranged to associate a plurality of synchronization constructs with information relating to address locations in the object-based system, the plurality of synchronization constructs including the first synchronization construct, wherein at least one of the plurality of synchronization constructs is associated with more than one address location.

13. (Original) A computer program product for acquiring access to an object as recited in claim 12 wherein the computer code for identifying a first synchronization construct that is suitable for use granting access to the object further includes:

computer code for obtaining a hash value from the memory address value, wherein the computer code indexing into the data structure using the at least part of the memory address value includes computer code for indexing into the data structure using the hash value.

14. (Original) A computer program product for acquiring access to an object as recited in claim 13 wherein the information relating to address locations includes hash values of the address locations, and the data structure is a hash table.

15. (Original) A computer program product for acquiring access to an object as recited in claim 14 wherein the first synchronization construct is a lock and the plurality of synchronization constructs are locks, and the computer code for associating the first synchronization construct with the object includes computer code for locking the object using the first synchronization construct.

16. (Original) A computer program product for acquiring access to an object as recited in claim 11 wherein the computer-readable medium is one selected from the group consisting of a hard disk, a floppy disk, a data signal embodied in a carrier wave, a tape drive, an optical drive, and a CD-ROM.

17. (Currently Amended) A data structure for use in a multi-threaded, object-based computing system, the data structure being arranged to be stored in a memory of the computing system, the data structure comprising:

a plurality of index values; and

a plurality of lock identifiers, each lock identifier included in the plurality of lock identifiers being arranged to identify an associated lock in the computing system, wherein the plurality of lock identifiers is associated with corresponding index values included in the plurality of index values;

wherein the associated lock is held by a thread, and the associated lock can be released **such that an object associated with the lock is not locked by the thread** in response to a determination that a requesting thread is also the thread that is holding the associated lock.

18. (Original) A data structure according to claim 17 wherein at least one lock identifier included in the plurality of lock identifiers is associated with more than one index value included in the plurality of index values.

19. (Original) A data structure according to claim 18 wherein the data structure is arranged to be accessed by a thread using a value corresponding to a first index value included in the plurality of index values to identify a suitable lock for use in locking a first object included in the computing system.

20. (Original) A data structure according to claim 17 wherein the plurality of index values are hash values of address locations associated with the computing system.

21. (Currently Amended) A computer-implemented method for acquiring access to an object in an object-based system, the method comprising:

providing a thread that attempts to acquire access to an object;

identifying a memory address value associated with the object;

identifying a first synchronization construct that is suitable for use in granting access to the object;

determining ~~whether~~ **that** the first synchronization construct is unavailable;

when it is determined that the first synchronization construct is unavailable, determining ~~whether~~ **that** the thread that is attempting to acquire access to the object is already associated with the object; and

when it is determined that the thread that is attempting to acquire access to the object is already associated with the object, releasing the association **such that the thread is not associated with the object**.

22. (Currently Amended) A computer program product for acquiring access to an object in an object-based system, the computer program product comprising:

computer code for providing a thread that attempts to acquire access to an object;

computer code for identifying a memory address value associated with the object;

computer code for identifying a first synchronization construct that is suitable for use in granting access to the object, wherein the first synchronization construct is identified using at least part of the memory address value;

computer code for determining whether the first synchronization construct is unavailable;

computer code for, when it is determined that the first synchronization construct is unavailable, determining whether the thread that is attempting to acquire access to the object is already associated with the object;

computer code for, when it is determined that the thread that is attempting to acquire access to the object is already associated with the object, releasing the association **such that the thread is not associated with the object**; and

a computer-readable medium that stores the computer codes.

23. (New) A method for acquiring access to an object as recited in claim 1, wherein determining that the first synchronization construct is unavailable includes determining that the first synchronization construct is in use.

24. (New) A computer-implemented method for acquiring access to an object as recited in claim 21, wherein determining that the first synchronization construct is unavailable includes determining that the first synchronization construct is in use.

25. (New) A computer program product according to claim 22 further comprising:

computer code for, when it is determined that the first synchronization construct is available, associating the first synchronization construct with the object.